S. Kuwabara, R. Ohbuchi and T. Furuya, "Query by Partially-Drawn Sketches for 3D Shape Retrieval,"
2019 International Conference on Cyberworlds (CW), Kyoto, Japan, 2019, pp. 69-76. doi: 10.1109/CW.2019.00020

Query by partially-drawn sketches for 3D shape retrieval

Shutaro Kuwabara
Computer Science and Engineering Department
University of Yamanashi
Kofu, Japan
e-mail: t13cs016@gmail.com

Ryutarou Ohbuchi
Computer Science and Engineering Department
University of Yamanashi
Kofu, Japan
e-mail: ohbuchi@yamanashi.ac.jp

Takahiko Furuya
Computer Science and Engineering Department
University of Yamanashi
Kofu, Japan
e-mail: ohbuchi@yamanashi.ac.jp

***Abstract*— Hand-drawn sketch is a powerful modality to query 3D shape models. However, specifying a detailed 3D shape by a sketch on the first try without reference (i.e., 3D model or real object) is difficult. In this paper, we aim at a sketch-based 3D shape retrieval system that tolerates coarsely drawn or incomplete sketches having small number of strokes. Such a system could be used to start a sketch-retrieve-refine interactive loop that could lead to a 3D shape having required shape details. Proposed algorithm uses deep feature embedding into common feature embedding space to compare sketches and 3D shape models. To handle coarse or incomplete sketches, a sketch, which is a sequence of strokes, is augmented by removing stroke for training a pair of DNNs to extract sketch features. A sketch feature is a fusion of an image based feature extracted by a convolutional neural network (CNN) and a 2D point sequence feature extracted by using a recurrent neural network (RNN). Embedding of 3D shape feature and the sketch feature is learned by using triplet loss. Experimental evaluation of the proposed method is performed using (simulated) incomplete sketches created by removing part of their strokes. The experiments show that sketch stroke removal augmentation significantly improved retrieval accuracy if queried by using such incomplete sketches.**

Keywords— 3D shape retrieval, sketch-based retrieval, deep metric learning, triplet network, recurrent neural network, 2D convolutional neural network.

I. INTRODUCTION

Hand-drawn sketch can be a powerful modality to query 3D shape models. We don't need to possess 3D shape models close enough to retrieval target. We don't have to know label of the target shape. And a sketch allows, albeit in 2D, for potentially detailed specification of shape, not just a broad category name, of the 3D models desired. However, we rarely have a concrete and detailed 3D shape in mind when we set about to sketch a query. We know what exact shape we wanted after the retrieval. However, we often don't know what we want when we start the process of often exploratory search. It is an iterative and interactive query refinement loop that brings out the shape details we want.

To support such an exploratory mode of search, a 3D shape retrieval system that supports a search with a not-so-well-defined query is needed. Such an exploratory search should most likely support multiple query modalities including text, 2D sketch, 2D photograph, and 3D shape model. For example, a search would start with a text query that specify a semantic class. Then, the iteration using progressively refined 2D hand-drawn sketches narrows down the shape to be retrieved. There could be another step, e.g., by using one of (or some of) the retrieved 3D model(s) as queries to retrieve final candidates.

Toward the goal of such an exploratory 3D shape search system having multiple query modalities, we explore in this paper a hand-drawn 2D sketch-based 3D shape retrieval system that supports partially drawn or coarse sketches as query. While there have been many studies on sketch-based 3D model retrieval, all such system that we know of accepts final, or completed sketch as their query input. If incomplete sketches are given, retrieval accuracy of such systems would significantly degrade.

In this paper, we propose and evaluate an algorithm for sketch-based 3D shape retrieval that assumes as its input incomplete or partially drawn sketches. Multimodal comparison of sketches and 3D shape models is handled by deep metric learning using Triplet Network [1], which creates a common feature embedding space (CFES). To improve features extracted from an incomplete sketch, the sketch is treated both as a series of 2D images whose stroke complexity gradually increases, and as a series of pen strokes that are trajectories of 2D points. Sketch image feature extracted by a convolutional neural network (CNN) and sketch stroke sequence feature extracted by using a recurrent neural network (RNN) are combined into a fused sketch feature via a shallow neural network (NN). The sketch feature extraction CNN is trained by using a data augmentation method that simulates unfinished sketches by removing sketch strokes. (The sketch stroke sequence feature does not require augmentation.) The fused sketch feature and 3D shape feature are transformed by a feature

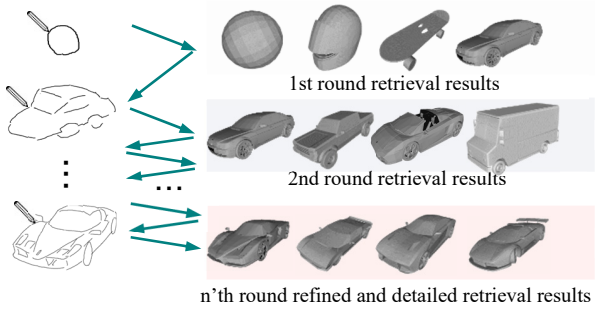


Fig. 1. Our ultimate goal: Exploratory and interactive "sketch-retrieve-refine" loop for coarse-to-fine sketch-based 3D model retrieval.

S. Kuwabara, R. Ohbuchi and T. Furuya, "Query by Partially-Drawn Sketches for 3D Shape Retrieval,"
2019 International Conference on Cyberworlds (CW), Kyoto, Japan, 2019, pp. 69-76. doi: 10.1109/CW.2019.00020

embedding network into features embedded in the CFES for direct comparison.

Experimental evaluation showed that proposed algorithm trained with the sketch stroke removal data augmentation produced good retrieval accuracy even for coarse sketch consisting of small number of strokes. Also, combining image feature and stroke sequence feature significantly improved retrieval accuracy compared to the case where only one of the features is used.

Contribution of our work can be summarized follows;

- Proposal of a sketch-based 3D shape retrieval algorithm that can handle partially drawn or coarse sketches. To do so, the system uses a sketch feature that combines image-based and stroke-sequence based features. The feature extractor is trained using a sketch data augmentation that removed strokes from sketch stroke sequence to simulate partially drawn sketches.
- Multimodal comparison of heterogeneous data objects, sketch and 3D shape, is done by using common feature embedding space induced by Triplet Network [1].
- Experimental evaluation of the above approaches that showed the proposed approaches are effective for 3D shape retrieval using coarsely drawn sketches.

The rest of this paper is structured as follows. We review, in the next section, work related to the proposed algorithm. We then describe the proposed algorithm in Section III, followed by experiments and results in Section IV. We conclude the paper in Section V.

II. Previous Work

We briefly review previous work on sketch-based 3D model retrieval methods.

A. Sketch-based 3D model retrieval

Retrieval of 3D models based on hand-drawn 2D sketches has been a difficult problem for several reasons. First, a sketch, be it a sketch image or a sequence of pen strokes that produces the sketch, is of different data type, or in a different data domain, than a 3D shape model. Comparison requires a mean to bridge the domain gap. Second, a sketch can be highly variable. It varies significantly due, for example, to semantic abstraction, incomplete knowledge of or ambiguities in the 3D shape to be drawn, drawing technique, drawing style preference, drawing medium and tool used, time allotted for drawing.

Eitz et al. [2] tried to bridge the domain gap by rendering a 3D model into sketch-like images from multiple viewpoints. The comparison between a sketch image and a 3D model is made in the domain of sketch-like images by using hand-crafted image features. Furuya et al. [3] tried to bridge the domain gap by employing a cross-modal manifold learning so that 3D shape features of 3D models can be compared with sketch images on a manifold constructed from sketch and 3D model data.

Recent advent in deep neural network (DNN) made comparison between heterogeneous data domains easier. Wang et al. [4] used Siamese network and contrastive loss to compare sketch-like rendering of 3D models with sketches. The features are embedded into a learned CFES for direct feature to feature similarity computation. The system achieved good retrieval accuracy given finished sketches. However, as the system assumed finished sketches as its input, retrieval accuracy would degrade significantly if the sketches are unfinished and have fewer number of strokes.

In Shape2Vec by Tasse et al. [5], 3D shape feature, sketch feature, and photographic image feature are all embedding into a CFES for cross-domain comparison. The common embedding space is formed as a feature space of words induced by Word2Vec [5] from a text corpus such as a Wikipedia snapshot. Features from 3D models (which is view-based using DNN), sketches, and photographs are embedded in the common feature space so that features of the same category are closer regardless of their original data domains. Common feature embedding space is a powerful approach that allows comparison of data from heterogeneous domains. Our proposed method employs CEFS to compare sketches with 3D models.

Majority of the work in sketch-based retrieval (or sketch image classification) are based on "coarse" sketches, that is, finer differences isn't an issue so far as their object classes match. In many practical retrieval tasks, however, details do matter. The user often wants a pair of shoes with a specific detail (e.g., toe shape or heel height), not just a shoe. Qian et al. [7] tackled this issue by using a DNN structure called Triplet Network [1] to retrieve images of objects based on sketches having specific details. This method also uses a CFES, and it is induced by training by using triplet loss. The Triplet Netowrk is trained by using triplets of features ($f_{reference}$, $f_{positive}$, $f_{negative}$). A set of triplets induces a feature space where $f_{reference}$ and $f_{positive}$ are close while $f_{reference}$ and $f_{negative}$ are distant. Note that this training does not explicitly use labels. The triplet simply specifies ranking among a pair of similarities. We employ Triplet Network to induce the CFES among 3D shape features and sketch features.

While a sketch is most commonly seen as a 2D image, it could also be a sequence of strokes or points if so recorded. Ha, et al. [8] in Sketch-RNN regarded a sketch as a sequence of strokes, each of which is in turn a sequence of points, for sketch recognition. A stroke sequence of a sketch is fed into an RNN to extract a stroke-based sketch feature. Xu et al. [9] in Sketchmate also treated a sketch as a point to retrieve sketches. Sketchmate treated a sketch both as a sequence of pooints as well as images. An image of a sketch is fed to a 2D CNN to extract 2D image feature. At the same time, a sequence of 2D points of a sketch is fed to an RNN to produce a stroke-based feature. These two features are combined using a neural network for retrieval. To speed up search, the combined feature is the hasned, using a DNN into a binary domain for fast comparison using Hamming distance. Our approach in this paper also use a combination of image based feature and stroke (2D point) sequence based feature for sketches to retrieve 3D shape models.

All the previous methods that we know of for sketch-based 3D shape retrieval considered a query sketch as finshed product. However, this assumption is not accurate if we were to support exploratory retrieval of the sketch-

S. Kuwabara, R. Ohbuchi and T. Furuya, "Query by Partially-Drawn Sketches for 3D Shape Retrieval,"
2019 International Conference on Cyberworlds (CW), Kyoto, Japan, 2019, pp. 69-76. doi: 10.1109/CW.2019.00020

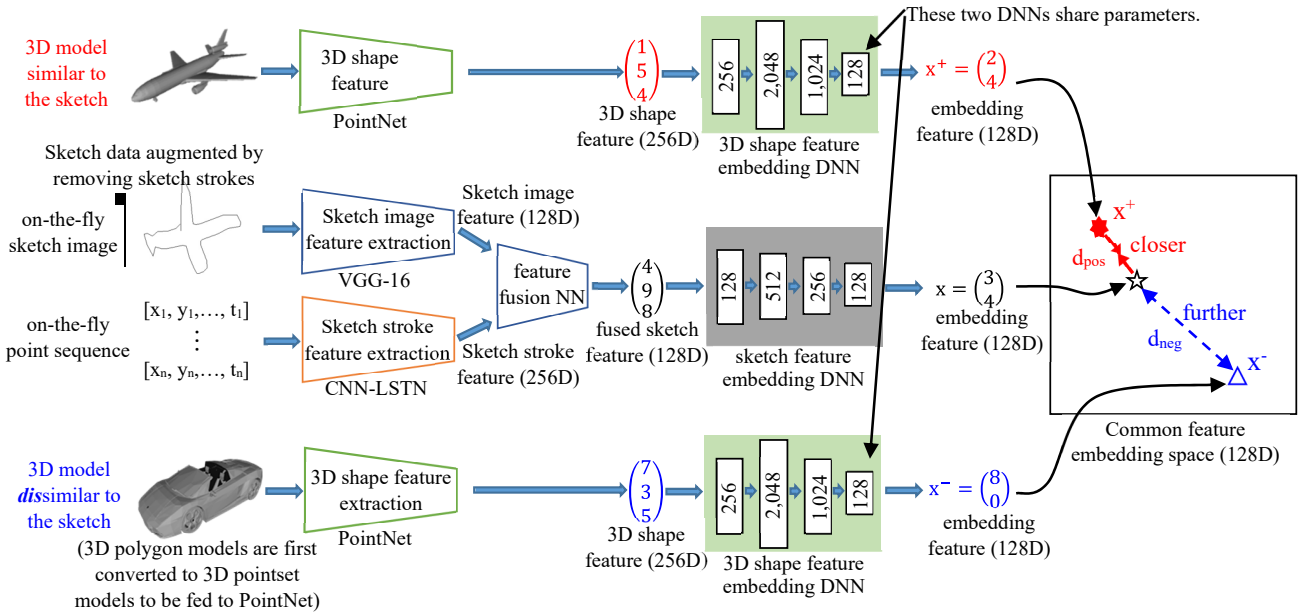


Fig. 2. The proposed method compares two heterogeneous datatypes, 3D shape and 2D sketch, in a common feature subspace induced by deep metric learning. Features of sketches are extracted, by using a pair of deep neural networks (DNN), from two data sources, that are, the 2D sketch image and the 2D point sequence of sketch strokes. These two features are combined into a fused sketch feature for the embedding. Features for 3D shapes to be embedded are extracted from point-set representation of the 3D model.

retrieve-refine loop. Our proposed method compares 3D shape feature and sketch feature in the CFES learned by Triplet Network. The sketch feature is a combination of sketch stroke sequence feature and sketch image feature, latter of which is trained by using data augmentation that simulates unfinished, or partially drawn, sketches. (As noted before, the sketch stroke sequence feature does not require explicit augmentation.)

B. 3D shape feature

Many classical approaches that employ hand-crafted 3D shape features, for example, [10], [11], [12], [13], exist. A good survey of work in this class can be found in [14]. With recent advance in neural network, many end-to-end neural network have been proposed for 3D shape recognition.

Voxel representaion is used in [15], [16], and others. Voxel representation of 3D shape is a natural extension of 2D image as 2D array of pixels. Many 2D CNNs can be extended naturally to accommodate voxels, a 3D array of scalar values. Voxel representation has some disadvantages, however. Due to limitation in DNN complexity, voxel size is limited so the shape details are often lost when 3D shape is converted to low-resolution voxel representation. It is also difficult to realize full 3D rotation invariance. Another popular approach is view-based. 3D shape models are rendered from multiple viewpoints into a set of images and 2D CNN for images are applied to each image (for example [17] and [18]). Rendered images could capture details to achieve high retrieval accuracy, but full rotation invariance is still difficult to achieve.

A more recent trend is to use point set representation of 3D shape, as in PointNet [19], PointNet++ [20], and SO-net [21]. PointNet achieves partial invariance to rotation via Spatial Transformer network embedded in it. Another trend is to extend DNN to handle polygonal representation [22]. 3D point set representation of 3D shape is a "minimal" representation without connectivity or topology, somewhat like raw pixels in 2D images. Point-based representation is spatially efficient, in that there is no point allocated for empty space. We employ PointNet for 3D shape feature extraction.

III. Proposed Method

Proposed method employs CFES to compare 3D shape feature of 3D models in the database with sketch feature extracted from a query sketch (See Fig. 2). Embedding of 3D shape feature and sketch feature is handled by using a pair of 4 layer fully connected NNs trained by using triplet loss [7]. Fused sketch feature is a combination of sketch image feature extracted by a CNN and sketch stroke sequence feature extracted by an RNN. Fusion of the two sketch features is performed either by simple vector concatenation or by a NN. A 3D shape feature is extracted from the 3D point set representation of a 3D shape by using PointNet [19]. The following sections will describe each of these components in more details.

A. Triplet Network for Feature Embedding

Embedding of 3D shape features from the PointNet (256D) and fused sketch features (128D) into the CFES having 128D is done by using a Triplet Network [1], a network for deep metric learning without using explicit labels. During training, the Triplet Network uses two DNNs, one for embedding fused sketch feature and another for embedding 3D shape feature. The 3D shape feature

S. Kuwabara, R. Ohbuchi and T. Furuya, "Query by Partially-Drawn Sketches for 3D Shape Retrieval,"
2019 International Conference on Cyberworlds (CW), Kyoto, Japan, 2019, pp. 69-76. doi: 10.1109/CW.2019.00020

embedding DNN embeds a 3D shape feature similar to the sketch ("positive" 3D model), as well as a 3D shape feature less similar to the sketch than the "positive" 3D model ("negative" 3D model). Thus, the dataset for training the network is a set of triplets (f_{sketch}, $f_{positive\_3D}$, $f_{negative\_3D}$), in which f_{sketch} is the fused sketch feature, $f_{positive\_3D}$ is the 3D shape feature for the positive 3D model, and $f_{negative\_3D}$ is the 3D shape feature for the negative 3D model. Training is done so that the following loss function is minimized.

$$lossT = \arg\max(0, m + d_{pos} - d_{neg}) \quad (1)$$

where d_{pos} is the distances between f_{sketch} and $f_{positive\_3D}$, and d_{neg} is the distance between f_{sketch} and $f_{negative\_3D}$. The margin m is set at 0.2 for the experiments below.

Table 1 and Table 2 show detailed network architectures of the feature embedding networks. L2 normalization of feature is applied at the output layers of respective embedding network. These embedding networks are rather shallow having only 4 layers, compared to the 23 layers for the deepest path of the proposed method depicted in Fig. 1. Such a shallow network appears reasonable since input features for sketch and 3D shape are already refined.

Input dimensionalities of the embedding NNs are chosen to match those of PointNet feature (256D) and fused sketch feature. Output dimensionalities of the embedding networks must equal the dimensionality of the CFES. We chose the dimensionality of the CFES at 128D for the experiments described below.

Note that the dimensionality of fused sketch feature, and thus the input dimensionality of the sketch feature embedding NN, depends on the sketch feature fusion method used. It will be 384D if the fusion is done by simple concatenation, and 128D if the fusion is done by sketch feature fusion NN described below.

TABLE 1. SKETCH FEATURE EMBEDDING NN STRUCTURE

Layer	# neurons	Activation	Dropout
Input	128	-	-
Layer1	512	ReLU	0.5
Layer2	256	ReLU	0.5
Output	128	-	-

TABLE 2. 3D FEATURE EMBEDDING NN STRUCTURE

Layer	# neurons		Activation	Dropout
Input	256		-	-
Layer1	2,048		ReLU	0.5
Layer2	1,024		ReLU	0.5
Output	128		-	-

B. Sketch Feature Extraction

As described before, a sketch feature is a combination of sketch image feature and sketch stroke sequence feature. We compare two combination methods, simple vector concatenation and a shallow neural network that performs feature refinement and dimension reduction.

1) Sketch Image Feature CNN (SIF-CNN)

To extract an image feature, a sketch image, which is generated as each complete stroke is added, is given to the Sketch Image Feature CNN (SIF-CNN). The SIF-CNN is VGG-16 [11] implemented as a part of Keras library. The numbers of neurons of the last three fully connected (FC) layers are changed from (4,096, 4,096, 1,000) to (256, 128, 75). The number of neurons of the last layer is set to 75 to match the number of classes of the training dataset. The SIF-CNN is pre-trained by using natural images contained in ImageNet dataset. Then the last 3 layers are fine-tuned by using sketch image classification task. After the training, activation of next to the last layer is used as 128D feature vector of sketch images.

For the fine tuning, to make the SIF-CNN robust against unfinished sketches having smaller number of strokes, data augmentation is done. The augmentation increased the number of images by the factor four. Details of the augmentation is described below.

2) Sketch Stroke Feature RNN (SSF-RNN)

Sketch stroke feature is extracted from 2D trajectory of points by using Sketch Stroke Feature RNN (SSF-RNN). SSF-RNN is modeled loosely after the encoder part of Sketch-RNN [8]. The SSF-RNN takes a sequence of 5D vectors $(\Delta x, \Delta y, p_1, p_2, p_3)$, each of which contains 2D displacement $\Delta x, \Delta y$ and three attributes. The attributes p_1, p_2, p_3 indicate, if "1", "pen down", "pen up", and "end of sketch", respectively.

The SSF-RNN is configured as three CNNs followed by two LSTMs (Table 3). The sequence of $(\Delta x, \Delta y, p_1, p_2, p_3)$ is processed first by the 3 layers of 1D CNN, whose output enters the LSTM having 2 units. Output of the LSTM are refined by using three layers of fully connected network (FCN) followed by a softmax classification layer. The LSTM is implemented using CuDNNLSTM for efficiency.

SSF-RNN is trained by using sketch classification task using categorical loss. No data augmentation is done for the training. After training the network using cross entropy loss with a 75 class sketch dataset, activation of 256 neurons of the next to the last layer is used as a 256D sketch stroke sequence feature.

TABLE 3. STRUCTURE OF SKETCH STROKE FEATURE RNN

Layers	Description
Input	1D sequence of 5D vectors
1D convolution	48D, kernel size 5, batch normalize
1D convolution	64D, kernel size 5
1D convolution	96D, kernel size 3
LSTM	128D
LSTM	128D
Fully connected	256D, return sequence
Fully connected	256D
Fully connected	75D
Softmax	-

3) Sketch Feature Fusion NN (SFF-NN)

Fusion of sketch image feature and sketch stroke feature is done either by vector concatenation or by a shallow fully connected NN called Sketch Feature Fusion NN (SFF-NN). Sketch image feature (128D) and sketch stroke sequence feature (256D) are concatenated to become 384D feature vector. If SFF-NN is employed, the concatenated vector then enters the SFF-NN to yield refined and dimension

reduced feature having 128D. The architecture of SFF-NN is shown in Table 4. SFF-NN is trained by using sketch image classification task, and the activation of the next to the last layer having 128 neurons is used as fused and refined sketch feature vector. As noted, number of neurons of the sketch feature embedding NN is adjusted to either 384D or 128D depending on the method of fusion used.

In Section IV, we will experimentally compare the two sketch feature fusion methods for their impact on accuracy.

TABLE 4. STRUCTURE OF THE SFF-NN

Layer	# neurons	Activation	Dropout
Input	384	-	-
Layer1	256	ReLU	0.5
Layer2	128	ReLU	0.5
Softmax	75	-	-

C. 3D Shape Feature DNN

3D shape feature of a 3D shape model is extracted by using PointNet [10]. PointNet accepts a set of 3D points, and performs recognition of, or, if so trained, segmentation of the point set. Thanks to the maximum pooling of point-wise features, 3D shape recognition results (and features) are unaffected by ordering of input 3D points.

As PointNet accepts 3D point set as its input, polygon based 3D shape models in the datasets are converted into 3D point sets. The conversion is done by sampling a 3D model with points randomly and uniformly placed on its polygonal surfaces. In the experiment described in this paper, 3D models are sampled by 2,048 points.

The PointNet is trained from scratch by a set of 3D shape models having 65 classes using cross entropy loss. (See next section for the 3D model dataset used.) After the training, activation of 256 neurons of the last layer before the softmax classification layer of the PointNet is used as 3D shape feature vector having 256D.

D. Training the Networks

Training of the proposed network is done part by part, in stages, to facilitate convergence and to keep cost of training concise. We first train SIF-CNN, SSF-RNN, and PointNet independently. We then train SFF-NN by using feature vectors generated by the trained SIF-CNN and SSF-RNN. Finally, we train the feature embedding network using the fused sketch features and 3D shape features.

1) Dataset

The dataset for training and testing requires both sketch data and 3D shape model data.

As the sketch data, we use 75,000 sketches of the Quick, Draw! dataset [12] having 75 classes, 1,000 sketch per class. A sketch of the Quick, Draw! dataset is a sequence of 2D points in which each point has pen up/down and end-of-sketch flag. The 75,000 sketches are split into 56,250 training set and 18,750 test set. Sketches in the Quick, Draw! dataset is coarse and simple, as they are drawn with the constraint of "draw until the system correctly recognizes what is drawn, or 20 seconds, whichever comes earlier". Consequently, sketches contain relatively small number of strokes. Fig. 3 show the histogram of stroke counts, whose mode 3.0 and averages 5.4.

Note that, while the SIF-CNN and SSF-RNN are trained using 75 categories of the sketch dataset for feature extraction, sketch (and 3D model) dataset used for retrieval experiment has only 65 categories. This is the result of matching (taking intersection of) classes in sketch dataset and 3D model dataset.

As the 3D model dataset, we use a set of 42,393 3D shape models defined as polygonal models. It is a union of ShapeNetCore55 dataset [12] and SHREC 2015 non-rigid retrieval track dataset [13]. ShapeNetCore55 contains mostly rigid models of automobiles, chairs, airplanes, etc., while SHREC 2015 non-rigid contains non-rigid (articulated or deformed) models of animals, human, etc. The set of 3D models is split into 33,914 model train set and 8,479 test set.

Fig. 3 shows examples of sketches and 3D models in corresponding categories from the databases.

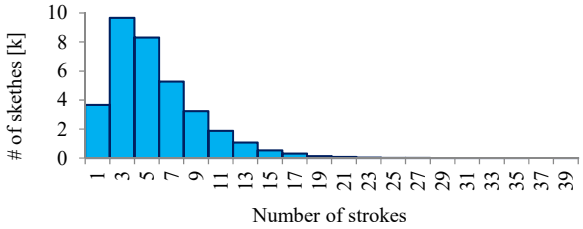


Fig. 3. Distribution of number of strokes of sketches in a subsampled Quick, Draw! dataset [12].

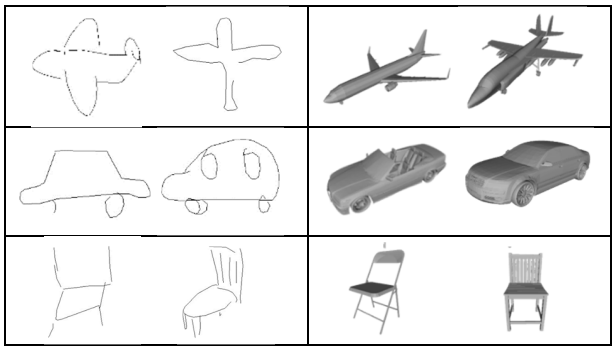


Fig. 4. Examples of sketches in the sketch dataset and corresponding 3D models in the 3D model dataset.

2) Training SIF-CNN:

We start with the VGG-16 network pre-trained by using natural images of the ImageNet database. We then fine tune the last 3 fully connected layers by using sketches. To train SIF-CNN, the sketch strokes sequence of a sketch is rendered into a 64×64 pixel image. For the proposed method, to increase robustness against unfinished or partially drawn sketches, we apply data augmentation that removes 1, 2, and 3 strokes from the original ("finished") sketch stroke sequence. For each sequence, strokes are removed from the tail (last) of the sequence. This data

S. Kuwabara, R. Ohbuchi and T. Furuya, "Query by Partially-Drawn Sketches for 3D Shape Retrieval,"
2019 International Conference on Cyberworlds (CW), Kyoto, Japan, 2019, pp. 69-76. doi: 10.1109/CW.2019.00020

augmentation increases the sketch image count by the factor 4 to 225,000 sketches. For comparison, we also trained the SIF-CNN using finished sketches only.

Fig. 4 shows examples of sketches of airplane and ship with and without sketch stroke removal. As the original "finished" sketches of the Quick, Draw! dataset have small number of strokes, removing 3 strokes leaves little.

The SIF-CNN training is run for 400 epochs, with batch size 1,024. The training converged after about 200 epochs. The set of CNN parameters having the best validation accuracy is used for sketch image feature extraction.

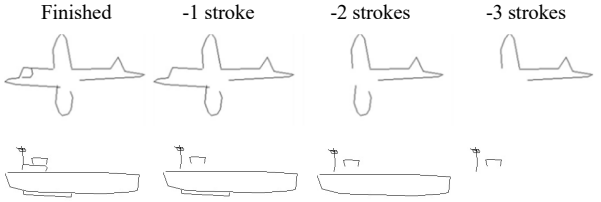


Fig. 5. Examples of Quick, Draw! sketches. Finished sketch (left) and sketches whose strokes are removed by 1, 2, or 3 strokes.

3) Training SSF-RNN

The RNN for sketch stroke feature is trained by using the stroke sequences of 75,000 sketches of the Quick, Draw! dataset. No data augmentation using stroke removal is applied to the training the SSF-RNN. SSF-RNN naturally deals with partially drawn sketches as it processes the sketch stroke-by-stroke (point by point, to be precise.)

The SSF-RNN training is run for 150 epochs with batch size 1,024. The training converged after about 100 epochs. The set of NN parameters having the best validation accuracy is used for stroke sequence feature extraction.

4) Training SFF-NN

After the SIF-CNN and SSF-RNN have finished training, their parameters are fixed to generate their respective feature vectors. Sketch image features and sketch stroke features are thus generated by using 56,250 sketches of the training set. With the factor of 4 data augmentation due to stroke removal, this amounts to 225,000 sketch features for training. Note that the SFF-NN is affected by the presence/absence of data augmentation by sketch stroke removal in training the SIF-CNN.

5) Training PointNet

PointNet is trained from scratch by using the 33,914 model train set of the 3D models. As mentioned above, these 3D models defined using polygonal representation are converted to 3D point set representation to be processed by PointNet. No data augmentation is used for 3D models.

6) Training Triplet Network

Training of the feature embedding triplet network is done by using a set of triplets (f_{sketch}, $f_{positive\_3D}$, $f_{negative\_3D}$). These triplets are generated randomly, based on their class labels, from 3D models and sketches. Given a sketch f_{sketch} having class c, a 3D model from class c becomes $f_{positive\_3D}$, while a 3D model from classes other than c becomes $f_{negative\_3D}$. For the experiments described below, we generated 100,000 triplets to train the triplet network. The number of triplets for embedding network training is fixed at 100,000 with or without the stroke removal data augmentation of sketches. Note that the data augmentation by sketch stroke removal (or lack of thereof) in SIF-CNN consequently affects training of the Triplet Network for data embedding and thus the CFES resulted.

IV. Experiments and Results

We experimentally evaluate (1) effectiveness of sketch features, that are, image feature, stroke sequence feature, and combined feature, and (2) effectiveness of training by using unfinished sketches. After all the training is done, we presented the system with a sketch from the test set, and computed precision and recall of the ranked list of 3D models retrieved based on the class labels attached to the sketch and retrieved 3D models.

We implemented the proposed algorithm using TensorFlow and Keras, and run the experiments on a PC with Intel Xeon E5-2680v2 CPU running Ubuntu 14.04LTS equipped with an Nvidia GeForce GTX1080 GPU.

A. Effectiveness of Combined Sketch Feature

This experiment compares accuracy of various sketch features. In this comparison, SIF-CNN for sketch image feature extraction is trained *without* stroke removal data augmentation of sketches. The evaluation is done using finished sketches (i.e., no strokes removed).

Table 5 lists retrieval accuracies in Mean Average Precision (MAP) [%]. In the table, "C+R (NN)" and "C+R" show the cases in which both sketch image feature extracted using CNN and sketch stroke feature extracted using RNN are used. The former with "NN" uses SFF-NN to fuse the two, while the latter simply concatenated the two features. The other two entries "R" and "C", respectively, show SSF-RNN only and SIF-CNN only cases.

As expected, using the two features fused by SFF-NN produced the highest accuracy. Between CNN and RNN, RNN performs significantly better if presented with finished sketches as queries.

TABLE 5. RETRIEVAL ACCURACY WHEN QUERIED USING FINISHED SKETCHES. (THE SYSTEM IS TRAINED *WITHOUT* STROKE REMOVAL DATA AUGMENTATION)

Feature	MAP [%]
C	23.8
R	38.9
C+R	40.2
C+R(NN)	**44.4**

B. Effectiveness of stroke removal augmentation

In this experiment, effect of training SIF-CNN by using data augmentation, that is, sketch stroke removal to simulate unfinished sketches, is evaluated.

This evaluation is done by using (simulated) unfinished sketches, in which last 0, 1, 2, or 3 strokes are removed from the stroke sequence. For comparison, NNs are trained with or without sketch stroke removal data augmentation. Cases

with data augmentation are marked "augmented", while cases without data augmentation are marked "not augmented".

Fig. 6 shows "averaged" retrieval accuracy for various sketch features with/without data augmentation. The average is over the number of removed strokes of the in the test set. As expected, "R+C(NN)(augmented)", which uses stroke removal augmentation, combined with NN-fused sketch feature, performs the best (MAP=0.447). Interestingly, when *evaluated* using stroke removed sketches, *sketch image feature* trained with data augmentation C(augmented) produced MAP=0.355. This accuracy is better than that of the sketch stroke feature R with MAP=0.301. (Recall that, in Table 5, when trained and evaluated *without* data augmentation, sketch stroke feature R outperformed sketch image feature C.)

Fig. 7 shows, for NN-fused sketch feature (R+C(NN)) and for sketch image feature only (C), retrieval accuracy evaluated with unfinished sketches simulated by removing last few strokes of query sketches. "0", "-1", "-2" or "-3" of the horizontal axis indicates strokes removed from the *test set* sketches. "0" corresponds to the completed sketch with all the strokes. As expected, accuracies drop as the strokes are removed. However, those trained with stroke removal data augmentation (solid lines) are more robust against loss of strokes in the test set, with smaller losses in accuracy.

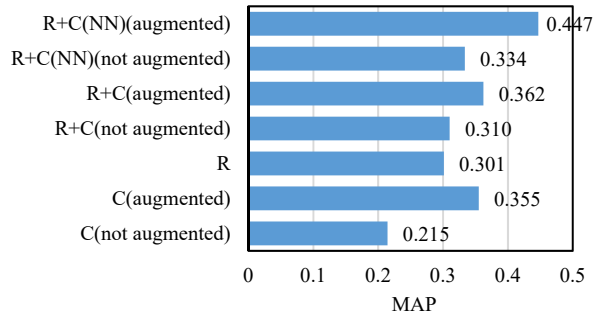


Fig. 6. Retrieval accuracy (MAP) averaged over number of strokes (0, 1, 2, and 3) removed from test set query sketches.

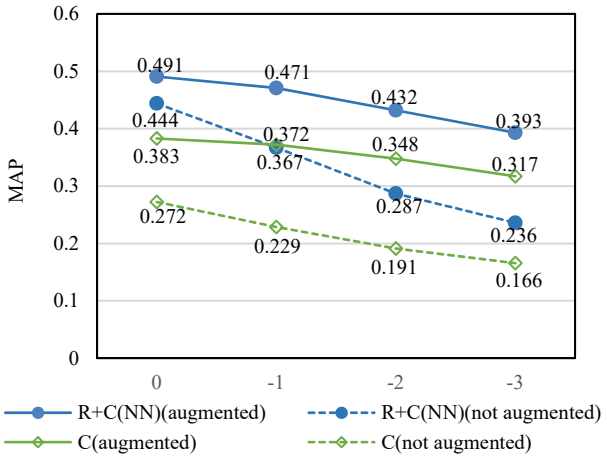


Fig. 7. Effect of removing strokes in *query* sketches to simulate unfinished sketches.

Fig. 8 shows examples of retrieval using sketches of an automobile and a train. Sketching a car starting from its characteristic wheel well quickly retrieved 3D models of automobiles. Positive contribution of fusing sketch-image and sketch stroke sequence feature is observed.

V. Conclusion and Future Work

Sketch based 3D models retrieval is a powerful modality in querying 3D shape models. However, when put to practical use, making a detailed sketch of a 3D shape that exists only in one's mind is not easy thing to do. It is a minority who could draw bicycle or automobile with enough accuracy and detail to retrieve a specific (not any) bicycle or automobile she/he wants. A 3D shape retrieval system that supports "exploratory" mode of search that allow coarse or unfinished sketch having small number of strokes is thus needed. The search would start with a coarse sketch, but the sketch and 3D shape retrieval results would be refined over interactive search iterations.

With such ultimate goal in mind, we proposed a sketch based 3D shape retrieval system that supports coarse, partially drawn sketches. The system extracts sketch feature from both sketch stroke sequence and sketch image sequence that are trained by using sketch dataset augmented by removing last few strokes of sketches. The sketch feature and 3D shape feature are compared in a common feature embedding space induced by deep feature embedding. Experimental evaluation showed that the data augmentation via sketch stroke removal significantly improved retrieval accuracy when unfinished sketches are given as queries.

In the future, we would like to find more accurate and less computationally expensive ways to induce common feature embedding space. We also need a better data augmentation method to handle partial and diverse sketches. We then need to develop and quantitatively evaluate a system that allows multiple query modalities (text, sketch, etc.) and interactive sketch-retrieve-refine loop. To that end, we have to establish datasets and evaluation protocol for such coarse-to-fine interactive search of 3D models.

References

[1] Hoffer E., Ailon N., "Deep Metric Learning Using Triplet Network," Springer LNCS, Vol. 9370, 2015.

[2] Eitz, R.M., Boubekeur, R., Hildebrand, T., and M, Alexa, "Sketch-Based Shape Retrieval," ACM TOG, 31（4）, No.31, 2012

[3] Furuya, T., Ohbuchi, R., "Ranking on cross-domain manifold for sketch-based 3D model retrieval." In Cyberworlds 2013, pp. 274–281, 2013.

[4] Wang R., Kang L., and Li Y., "Sketch-based 3D Shape Retrieval Using Convolutional Neural Networks," Proc. CVPR 2015, 2015.

[5] Tasse, F.P. and Dodgson, N., "Shape2Vec: Semantic-based descriptors for 3D shapes, sketches and images," ACM Trans. Graph., vol. 35, no. 6, Art. no. 208, 2016.

[6] Mikolov, T., Chen, K., Corradoc, G., and Dean, J. "Efficient estimation of word representations in vector space." ICLR Workshop. 2013.

[7] Qian Yu, et al., "Sketch Me That Shoe", In CVPR, 2016.

[8] Ha, D., Eck, D., "A Neural Representation of Sketch Drawings" arXiv preprint arXiv:1704.03477, 2017.

S. Kuwabara, R. Ohbuchi and T. Furuya, "Query by Partially-Drawn Sketches for 3D Shape Retrieval,"
2019 International Conference on Cyberworlds (CW), Kyoto, Japan, 2019, pp. 69-76. doi: 10.1109/CW.2019.00020

[9] Xu, P., Huang, et al. "Sketchmate: Deep hashing for million-scale human sketch retrieval.", In CVPR, 2018.

[10] Johnson, A.E., and Hebert, M. "Using spin images for efficient object recognition in cluttered 3D scenes", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 21, No. 5, pp.433-449, (1999),

[11] Osada, R., Funkhouser, T., Chazelle, B., and Dobkin, D. "Shape Distributions", *ACM Transactions on Graphics (TOG)*, Vol. 21, No. 4, pp. 807-832, (2002).

[12] Chen, D.Y., Tian, X.P., Shen, Y.T. and Ouhyoung, M., "On visual similarity based 3d model retrieval,", Computer GraphicsForum, vol. 22, no. 4, pp. 223–232, 2003.

[13] Wahl, E., Hillenbrand, U., and Hirzinger. G., "Surflet-pair-relation histograms: a statistical 3D-shape representation for rapid classification," *Proc. 3-D Digital Imaging and Modeling, 3DIM 2003*, pp. 474–481, (2003)

[14] ElNaghy, H., Hamad, S., and Khalifa, M. "TAXONOMY FOR 3D CONTENT-BASED OBJECT RETRIEVAL METHODS", *International Journal of Research and Reviews in Applied Sciences (IJRRAS)*, 14 (2), pp. 412-446, (2013).

[15] Wu, Z. et al. "3D ShapeNets: A deep representation for volumetric shape modeling. "*Proc. CVPR 2015*, 2015.

[16] Maturana D. and Scherer, S., "VoxNet: A 3D Convolutional Neural Network for Real-Time Object Recognition." *Proc. IROS 2015*, pp.922–928, 2015.

[17] Su, H., Maji, S., Kalogerakis, E., and Learned-Miller, E. "Multi-view Convolutional Neural Networks for 3D Shape Recognition," *Proc. ICCV 2015*, (2015).

[18] Shi, B., Bai, S., Zhou, Z., and Bai, X.: "DeepPano: Deep Panoramic Representation for 3-D Shape Recognition", *IEEE Signal Processing Letters*, 22(12), 2339–2343, (2015).

[19] Qi, C.R., Su, H., Mo, K., Guibas, L.J., "PointNet: Deep learning on point sets for 3d classification and segmentation," Proc. Computer Vision and Pattern Recognition (CVPR), IEEE, vol. 1, no. 2, p. 4, 2017.

[20] Charles R. Qi, Li Yi, Hao Su, Leonidas J. Guibas, PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space, *Proc. NIPS 2017*, pp. 5105–5114, 2017.

[21] Li, J., Chen, B.M., Lee, G.H., "SO-Net: Self-Organizing Network for Point Cloud Analysis" *Proc. CVPR 2018*.

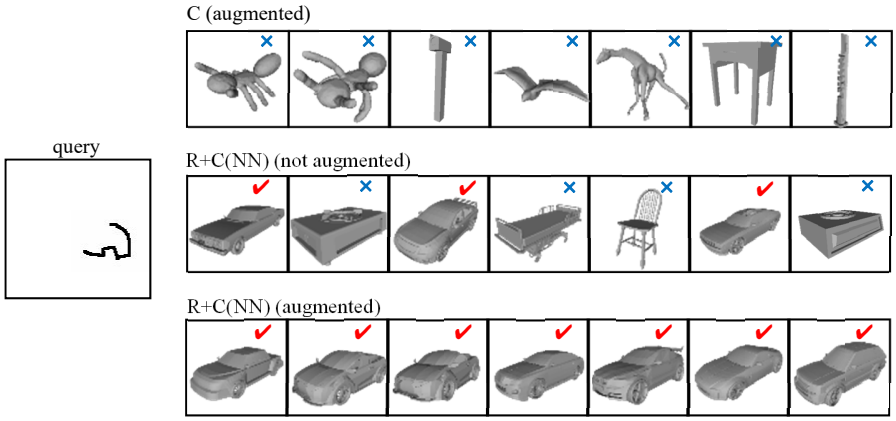
[22] Verma N., et al. "FeaStNet: Feature-Steered Graph Convolutions for 3D Shape Analysis" *Proc. CVPR 2018.*

[23] Simonyan, K., Zisserman, A., "Very Deep Convolutional Networks for Large-Scale Image Recognition", arXiv technical report, 2014.

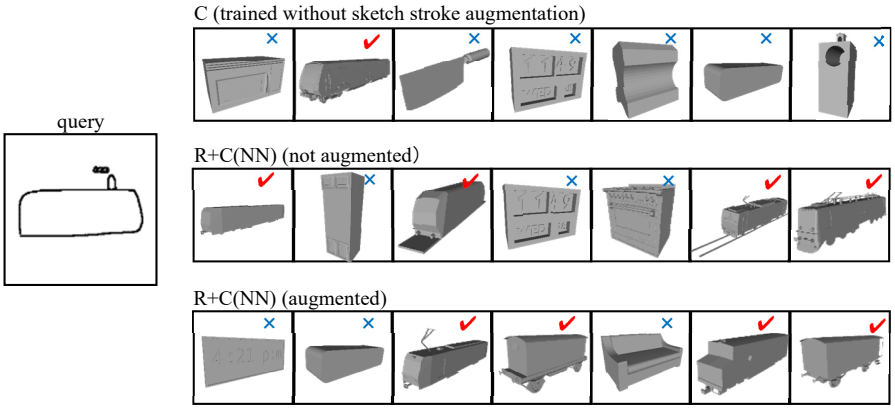
[24] Chang, A.X., et al., "ShapeNet: An Information-Rich 3D Model Repository", arXiv:1512.03012.

[25] Lian, Z., et al., "Non-rigid 3D Shape Retrieval," in Eurographics workshop on 3D object retrieval, 2015.

[26] Schuster, M., and Kuldip K.P., "Bidirectional recurrent neural networks." Signal Processing, IEEE Transactions on 45.11 (1997): 2673-2681.2.



(a) Success case: Sketching a wheel well, a very characteristic feature, retrieved automobiles.



(b) Failure case: Sketching a train car, even after completion, had many similar shapes in wrong class (sofa, etc.) retrieved.

Fig. 7. Example of sketch-based retrieval of 3D models.